PM5313

SPECTRA-622

DEVICE DRIVER ERRATA

Released

Production

Issue 2: November, 2001

Legal Information

Copyright

PMC-2000842 (R2)

Disclaimer

Trademarks

Patents

The technology discussed is protected by one or more of the following Patents:

U.S. Patent No. x,xxx,xxx y,yyy,yyy

Relevant patent applications and other patents may also exist.

Contacting PMC-Sierra

PMC-Sierra, Inc.
8555 Baxter Place
Burnaby BC Canada V5A 4V7

Tel: +1-604- 415-6000
Fax: +1-604-415-6200

Document Information: document@pmc-sierra.com
Corporate Information: info@pmc-sierra.com
Technical Support: apps@pmc-sierra.com
Web Site: http://www.pmc-sierra.com

Revision History

Issue No.	Issue Date	Details of Change
1	July 2000	Document created.
2	November 2001	Removed all previous errata addressed in Issue 1of Driver: ISR; Interrupt Enable Bit Not Clearing ISR; Auxiliary Interrupt Status Bits Not Clearing Memory leak due to callback defined as a NULL DLL OVERRIDE bit in 19.44 telecombus mode Concatenated payloads with normal mode initialization profile All errata from Issue 1 have been addressed in release 1.0 of the driver. Add driver errata 2.1.

Table of Contents

1 Introduction

The information in this document applies to Release 1.0 of the PM5313 SPECTRA-622 Device Driver and contains a complete list of all errata items associated with the PM5313 SPECTRA-622 Device Driver. Release 1.0 of the SPECTRA-622 device driver and Issue 2 of this errata document supersede all prior versions of the device driver and the errata document respectively.

1.1 Driver identification

The revision numbers of the driver files are indicated in the comment section of each file under Modification History. The driver release includes the following files:

Table 1 PM5313 SPECTRA-622 Device Driver Files.

Directory	Filename	File Version
example\	spe_app.c	1.8
example\	spe_app.h	1.4
example\	spe_debug.c	1.4
example\	spe_debug.h	1.3
inc\	spe_api.h	1.7
inc\	spe_defs.h	1.9
inc\	spe_err.h	1.6
inc\	spe_fns.h	1.6
inc\	spe_hw.h	1.9
inc\	spe_rtos.h	1.7
inc\	spe_strs.h	1.6
inc\	spe_typs.h	1.7
src\	spe_api1.c	1.11
src\	spe_api2.c	1.7
src\	spe_hw.c	1.7
src\	spe_isr.c	1.7
src\	spe_prof.c	1.6
src\	spe_rtos.c	1.9
src\	spe_stat.c	1.7
src\	spe_util.c	1.8
-	Makefile	1.9
-	software.lic	n/a
-	spectra_relnotes_r2.pdf	n/a

2 Functional Deficiencies

This section outlines the known functional deficiencies of the PM5313 SPECTRA-622 Device Driver as of the publication date of this document. For each deficiency, the known workaround and the operating constraints, with and without the workaround, are also described. Table 2 provides a summary of the functional deficiencies.

Please report any functional deficiencies not covered in this errata document to PMC-Sierra, Inc.

Table 2 Summary of Functional Deficiencies

Section	Discrepancy	Workaround
2.1	Driver Fails to report add bus parity error interrupts	Yes – edit code

2.1 Driver fails to report add bus parity error interrupts

2.1.1 Description

When configured in ISR mode, the driver can not service the add bus parity error interrupt (API1, API2, API3 and API4 in Section/Line Block Interrupt Status register 000BH). When configured in polling mode the driver services the interrupts correctly.

2.1.2 Workaround

The current implementation of the driver assumes that the add bus parity error interrupt (API1, API2, API3 and API4 in Section/Line Block Interrupt Status register 000BH) are covered by one of the master interrupt status bits within register 000BH (SPECTRA-622 Section/Line Block Interrupt Status register): CSPII or CRSII. This is incorrect and therefore code modifications must be made to the `isrSpectraGetStatusIO()`, `isrSpectraGetStatus()`, and `isrSpectraPollStatus()` functions located in `spe_isr.c`.

`isrSpectraGetStatusIO()` should be updated by removing the following code:

```
/* SPECTRA-622 ADD Bus Parity Interrupt Status */
addr = base + SPE_REG_ADD_BUS_PARITY_INT_STAT;
val = sysSpectraRead(addr);
pmask->ioApe[0] = val & SPE_BTMSK_IO_API1;
pmask->ioApe[1] = val & SPE_BTMSK_IO_API2;
pmask->ioApe[2] = val & SPE_BTMSK_IO_API3;
pmask->ioApe[3] = val & SPE_BTMSK_IO_API4;
```

`isrSpectraPollStatus()` should be updated by replacing the following code:

```
/* clear the isv mask */
sysSpectraMemSet(pmask, 0, sizeof(sSPE_MASK));
```

```
isrSpectraGetStatusIO (pddb, pmask);
```

with:

```
UINT1 * base,
      * addr;
UINT1 val;

/* clear the isv mask */
sysSpectraMemSet(pmask, 0, sizeof(sSPE_MASK));


/* SPECTRA-622 ADD Bus Parity Interrupt Status */
addr = base + SPE_REG_ADD_BUS_PARITY_INT_STAT;
val = sysSpectraRead(addr);
pmask->ioApe[0] = val & SPE_BTMSK_IO_API1;
pmask->ioApe[1] = val & SPE_BTMSK_IO_API2;
pmask->ioApe[2] = val & SPE_BTMSK_IO_API3;
pmask->ioApe[3] = val & SPE_BTMSK_IO_API4;

isrSpectraGetStatusIO (pddb, pmask);
```

`isrSpectraGetStatus()`, should be updated by replacing the following code:

```
baddr = pddb->baseAddr;

/* read the SPECTRA-622 Section/Line Block Interrupt Status register */
```

with:

```
baddr = pddb->baseAddr;

/* SPECTRA-622 ADD Bus Parity Interrupt Status */
addr = baddr + SPE_REG_ADD_BUS_PARITY_INT_STAT;
val = sysSpectraRead(addr);
pmask->ioApe[0] = val & SPE_BTMSK_IO_API1;
pmask->ioApe[1] = val & SPE_BTMSK_IO_API2;
pmask->ioApe[2] = val & SPE_BTMSK_IO_API3;
pmask->ioApe[3] = val & SPE_BTMSK_IO_API4;

/* read the SPECTRA-622 Section/Line Block Interrupt Status register */
```

2.1.3 Performance With Workaround

When configured in ISR mode or polling mode, the driver services an add bus parity error interrupt (API1, API2, API3 and API4 in Section/Line Block Interrupt Status register 000BH) correctly.

2.1.4 Performance Without Workaround

When configured in ISR mode, an add bus parity error interrupt (API1, API2, API3 and API4 in Section/Line Block Interrupt Status register 000BH) the driver will not service the interrupts and will lock the system in the ISR.

3 Documentation Deficiencies

We are not aware of any documentation deficiencies at this time. Please report any documentation deficiencies not covered in this errata to PMC-Sierra.